ประดิษฐ์ ภิญโญภาสกุล

มาตรฐานใหม่ของ รหัสแอสกี

ถึงเวลาแล้วหรือกับการเพิ่มขนาดของรหัสตัวอักษร รหัสไหนใครจะแน่กว่ากัน

ถึงเวลาแล้วที่จะมีการเพิ่มขนาดของรหัสแอสกี เพื่อรองรับตัวอักษรที่มีอยู่ทั้งหมดในโลก รวมทั้งตัวอักษรไทย

เพราะโลกของเราแคบลงทุกวันด้วยการติดต่อสื่อสาร และการคมนาคมที่ได้รับการพัฒนาอยู่ตลอดเวลา แต่เนื่องจากการที่คนทั้งโลกไม่ได้ใช้ภาษาเดียวกันทั้งหมด ข้อมูลข่าวสารต่าง ๆ จึงจำเป็นจะต้องมีการแปลไปเป็นภาษาอื่น ๆ ในวงการคอมพิวเตอร์ได้มีการออกแบบรหัสต่าง ๆ เพื่อใช้แทนตัวอักษรแต่ละภาษาขึ้นมากมาย ซึ่งก่อให้เกิดปัญหาการไม่มีมาตรฐานเดียวกันขึ้น อันจะเห็นได้จากรหัสภาษาไทยของเราเองก่อนที่เราจะมีรหัส ส.ม.อ. ใช้กัน

รหัสแอสกี

รหัสแอสกี (ASCII, the American Standard Code for Information Interchange) มีใช้อยู่ในทุกระบบคอมพิวเตอร์ แรกเริ่มเดิมทีมีขนาด 7 บิต ใช้แทนตัวอักษร a ถึง z ทั้งตัวเล็ก และตัวใหญ่, ตัวเลข, เครื่องหมายวรรคตอน, สัญลักษณ์บางอย่าง และคำสั่งควบคุมอุปกรณ์ต่าง ๆ รวม 128 (2^7) ตัว แต่คนทั่วไปมักรู้จักรหัส ASCII ในฐานะรหัสขนาด 8 บิตที่ใช้แทนตัวอักษร 256 (2^8) ตัว รหัสแอสกีได้รับการรับรองจากสถาบันมาตรฐานแห่งชาติของสหรัฐอเมริกา (ANSI, American National Standard Institute) ในปี พ.ศ. 2520 และองค์การมาตรฐานสากล (ISO, International Standards Organization) ได้กำหนดรหัสที่แทบจะไม่ต่างกันเลยขึ้นเป็นมาตรฐานในปีเดียวกัน เรียกว่า รหัส ISO 646 ซึ่งต่อมา ISO ได้เพิ่มขนาดของรหัส ISO 646 ขึ้นอีก 1 บิต เป็น 8 บิต เพื่อใช้แทนตัวอักษรอีก 128 ตัว รวมเป็น 256 ตัว รหัสใหม่นี้รู้จักกันในชื่อว่า รหัสละติน 1 ตัวอักษรที่เพิ่มขึ้นส่วนหนึ่งเป็นตัวอักษรที่ใช้ในภาษาของยุโรปหลายภาษา เช่น ฝรั่งเศส เยอรมัน และสเปน อันได้แก่ตัวอักษร é, ü และ ñ เป็นต้น

ประเทศต่าง ๆ ทั้งในอาฟริกา ตะวันออกใกล้ และตะวันออกไกล หลายประเทศ ได้รับเอารหัส ISO 646 มาใช้โดยทำการปรับปรุงให้ใช้กับตัวอักษรของ

ตนเอง และกำหนดเป็นรหัสมาตรฐานที่ยึดถือตามรหัส ISO 646 ขึ้น โดยได้รับความช่วยเหลือจากสมาคมผู้ผลิตคอมพิวเตอร์แห่งยุโรป (ECMA, the European Computer Manufacturers Association) และคณะกรรมการมาตรฐานอุตสาหกรรมแห่งชาติญี่ปุ่น ทั้งยังได้มีการพัฒนารหัสต่าง ๆ อย่างต่อเนื่องเพื่อใช้แทนตัวอักษรหลายพันตัวในระบบตัวอักษรของภาษาจีน, ญี่ปุ่น และเกาหลี ในประเทศไทยเองก็มีการออกแบบรหัสแอสกีภาษาไทยขึ้นใช้ตั้งแต่ปี พ.ศ. 2519 ซึ่งกว่าจะมีการประกาศใช้รหัส ส.ม.อ. เป็นมาตรฐานในปี พ.ศ. 2529 ก็มีรหัสภาษาไทยต่าง ๆ เกิดขึ้นมากมาย เฉพาะที่มีใช้บนเครื่องไมโครคอมพิวเตอร์ยี่ห้อต่าง ๆ ก็มีมากกว่า 20 แบบ

การที่ในหลาย ๆ ภาษามีรหัสที่ใช้แทนตัวอักษรของตนอยู่หลายรหัส ทำให้การติดต่อธุรกิจระหว่างชาติด้วยคอมพิวเตอร์เป็นไปได้อย่างลำบาก จึงเกิดความจำเป็นที่จะต้องมีการกำหนดมาตรฐานเพื่อใช้เป็นสากลร่วมกันขึ้น

ไบต์เดียวไม่เพียงพอเสียแล้ว

ในปี พ.ศ. 2526 ISO ได้เริ่มการพัฒนารหัสมาตรฐานใหม่ ในชื่อว่า ISO 10646 โดยทีแรกตั้งใจจะใช้ 2 ไบต์ (16 บิต) สำหรับ 65,536 ตัวอักษร ซึ่งเพียงพอสำหรับทุกตัวอักษรที่มีอยู่ทั้งหมดในปัจจุบัน คณะกรรมการของ ANSI ที่เข้าร่วมเป็นกรรมการในการพัฒนาได้ตั้งความหวังที่จะให้รหัสใหม่นี้เข้ากันได้ (compatible) กับรหัสมาตรฐานเดิม คือให้ 128 ตัวแรกเหมือนกันกับของรหัสแอสกีเดิม อย่างไรก็ตาม ได้เกิดปัญหาที่เห็นได้ชัดเจนขึ้น 2 ประการ ประการแรก รหัสมาตรฐานเดิมมีตัวควบคุม (control codes) รวมอยู่ด้วย เช่น ตัวควบคุมการปัดแคร่ (carriage return) และตัวควบคุมการป้อนบรรทัด (linefeed) เป็นต้น จึงได้เกิดข้อโต้แย้งกันขึ้นว่า ในรหัสใหม่จะยังคงตัวควบคุมเหล่านี้ไว้หรือไม่ ถ้าไม่คงไว้ ก็จะไม่เข้ากันกับรหัสมาตรฐานเดิม ในที่สุดได้ข้อตกลงว่าจะสำรองที่ 40 เปอร์เซ็นต์ จากทั้งหมด 65,536 ตัว ไว้สำหรับตัวควบคุม

ปัญหาประการที่สองคือ ปัญหาเกี่ยวกับระบบตัวอักษรของจีน, ญี่ปุ่น และเกาหลี ที่มีจำนวนมากที่สุด เนื่องจากระบบตัวอักษรของภาษาเหล่านี้ จะมีอยู่จำนวนมากทีเดียวที่เป็นตัวอักษรรูปภาพ (Pictographic characters) หรือตัวอักษรแบบคำ (Logographic characters) ซึ่งทั้งหมดได้รับการดัดแปลงมาจากตัวอักษรฮั่น (Han characters) ที่ประดิษฐ์ขึ้นในสมัยราชวงศ์ฮั่นของจีน ในจำนวนนี้มีหลายตัวที่นอกจากจะมีรูปร่างเหมือนกันแล้ว ยังมีความหมายเหมือนกันด้วย เช่น ตัวอักษรที่ใช้ในความหมายคำว่า "น้ำ" ในภาษาจีน, ญี่ปุ่น และเกาหลี เป็นตัวเดียวกัน เพื่อที่จะประหยัดเนื้อที่ คณะกรรมการจึงมีความเห็นว่าควรจะรวมตัวอักษรฮั่นเข้าไว้ด้วยกัน จะได้ไม่มีการกำหนดตัวอักษรที่เหมือนกันซ้ำกัน ส่วนตัวอักษรที่มีใช้เฉพาะในแต่ละภาษาจะมีการกำหนดแยกไว้อีกส่วนหนึ่งต่างหาก อันจะมีผลทำให้ประหยัดเนื้อที่ลงได้หลายพันตัวอักษร และจะทำให้เนื้อที่ 2 ไบต์มีขนาดเพียงพอที่จะแทนตัวอักษรของทุกภาษาได้ แต่อย่างไรก็ตาม ประเทศจีนยอมรับตามข้อเสนอนี้ แต่ญี่ปุ่น และเกาหลีไม่ยอมรับ

เพื่อแก้ปัญหาดังกล่าว คณะกรรมการจึงได้ตกลงที่จะกำหนดตัวอักษรของทุกภาษาเอาไว้ทั้งหมด ทำให้เนื้อที่ 2 ไบต์ไม่เพียงพอที่จะใช้แทนได้ คณะกรรมการใช้เวลากว่า 1 ปีในการหาข้อสรุปว่า จะเพิ่มขนาดเป็น 3 ไบต์หรือ 4 ไบต์ดี และในที่สุดตกลงที่จะใช้รหัสขนาด 4 ไบต์ (32 บิต) ซึ่งจะทำให้มีเนื้อที่สำหรับเก็บตัวอักษรถึง 4 พันล้าน (4,294,967,296) ตัว

4 ไบต์หาใช่ 4 เท่าเสมอไปไม่

มีหลายฝ่ายที่ไม่เห็นด้วยกับการใช้รหัสขนาด 4 ไบต์ ด้วยความเห็นที่ว่าการที่จะต้องส่ง 4 ไบต์แทน 1 ไบต์เดิมสำหรับแต่ละตัวอักษรจะไม่ทำให้เสียเวลาในการติดต่อสื่อสารเพิ่มขึ้นถึง 4 เท่า และขนาดของแฟ้มจะเพิ่มขึ้นถึง 4 เท่าด้วยกระนั้นหรือ ?

ในความเป็นจริงแล้วไม่เป็นเช่นนั้น เพราะ 3 ไบต์สูง (high bytes) ของตัวอักษรทุกตัว ของแต่ละภาษา จะเหมือนกัน จึงไม่จำเป็นที่จะต้องส่งไปทั้งหมด 4 ไบต์ เช่น ตัวอักษร c ในรหัสแอสกีก็คือ 01100011 ในรหัส ISO 10646 จะคล้ายกัน แต่เพิ่ม 3 ไบต์เข้าไปข้างหน้า (แต่ละไบต์เป็น 00100000) ได้เป็น 00100000001000000010000001100011

สาเหตุที่กำหนดให้ 3 ไบต์แรกเริ่มต้นจาก 00100000 ซึ่งแทนตัวอักษรเว้นวรรคในรหัส 8 บิต เนื่องจากตัวอักษร 00000000 ถึง 00011111 ในรหัส 8 บิตเดิมเป็นตัวควบคุม และคณะกรรมการได้ตั้งวัตถุประสงค์ไว้แต่แรกแล้ว ที่จะไม่กำหนดตัวอักษรใด ๆ ที่อาจทำให้เกิดการตีความหมายผิดเป็นตัวควบคุมได้

กรรมการหลายท่านได้เสนอวิธีการในการส่งข้อมูลด้วยขนาดข้อมูลที่เล็กลง เช่น โมเด็มตัวส่ง อาจส่งข้อมูลขนาด

4 ไบต์ตัวหนึ่งไปยังโมเด็มตัวรับเพื่อบอกว่า ต่อจากนี้จะเป็นตัวอักษรที่ขึ้นต้นด้วย 3 ไบต์แรกเป็น 00100000 ทั้งหมด ต่อจากนั้นจึงส่งเพียงไบต์ต่ำสุด (least significant byte) ของตัวอักษรแต่ละตัวไปยังโมเด็มตัวรับ เรียกวิธีการในการส่งข้อมูลแบบนี้ว่า วิธี Compaction ซึ่งเป็นคนละอย่างกับวิธี Compression โดยหลักการเดียวกันนี้สามารถใช้ได้กับทุกภาษา เพราะตัวอักษรของแต่ละภาษาจะมี 3 ไบต์สูงเหมือนกันทั้งหมด

แต่มีกลุ่มคนอีกกลุ่มหนึ่งได้แสดงความไม่เห็นด้วยกับวิธีดังกล่าว โดยให้เหตุผลว่า วิธีดังกล่าวจะก่อให้เกิดความยุ่งยากหลายอย่าง ควรที่จะส่งตัวอักษรที่มีขนาดเดียวไปเสมอมากกว่า จึงได้เกิดการแบ่งแยกออกเป็น 2 มาตรฐาน

รหัส Unicode

ในปี พ.ศ. 2530 กลุ่มคน 3 คน ซึ่งประกอบด้วย Joe Becker และ Lee Collins จากศูนย์วิจัย Xerox Palo Alto และ Mark Devis จากบริษัท Apple ได้ร่วมมือกันพัฒนารหัสที่ง่ายกว่ารหัส ISO 10646 ขึ้น ซึ่งจะยังคงใช้แทนตัวอักษรทุกตัวที่จำเป็นได้ทั้งหมด โดย Becker และ Collins ได้เคยร่วมงานกันพัฒนารหัส 16 บิตบนเครื่องคอมพิวเตอร์ Xerox Star ซึ่งเป็นต้นแบบของเครื่องแมคอินทอชมาก่อนเป็นเวลากว่าสิบปี ซึ่งเขาทั้งสองได้กล่าวว่า เวลาดังกล่าวทำให้พวกเขาได้ค้นพบข้อดีและข้อเสียของรหัสดังกล่าวเป็นอย่างดี

Becker และ Collins ได้ปรึกษาปัญหาร่วมกันกับ Devis ซึ่งเป็นหัวหน้ากลุ่มส่วนชุดคำสั่งระบบสากลของบริษัท Apple ในอันที่จะพัฒนารหัสสำหรับทุกภาษาขึ้น ซึ่งในขณะนั้น การสนับสนุนภาษาต่าง ๆ บนเครื่อง Apple เป็นเพียงการจัดเตรียมฟอนต์ของแต่ละภาษาไว้เท่านั้น ในขณะที่ทาง Xerox ได้ริเริ่มที่จะสร้างพื้นฐาน แทนการออกแบบเครื่องและระบบปฏิบัติการเป็นของตนเองเช่นที่เคยเป็นมาแต่ก่อน

Becker, Collins และ Devis จึงได้ร่วมกันตั้งกลุ่มทำงานเพื่อกำหนดรหัสมาตรฐานใหม่ขึ้น และให้ชื่อรหัสว่า รหัส Unicode (Unique, universal, and Uniform character encoding) ซึ่งในภายหลังได้มีตัวแทนจากบริษัทต่าง ๆ หลายบริษัท เข้าร่วมในการวางมาตรฐาน อันได้แก่ บริษัท Metaphor Computer Systems, the Research Libraries Group. บริษัท Microsoft, Sun, Adobe, Claris, Next และบริษัท Pacific Rim Connections ซึ่งกลุ่มผู้พัฒนารหัส Unicode นี้ ต่อมาได้เข้าร่วมเป็นคณะกรรมการพิจารณารหัส ISO 10646 ด้วย เพื่อเสนอแนวทางของรหัส Unicode ให้กับคณะกรรมการจากที่อื่น ซึ่งในจำนวนนี้มี บริษัท IBM, Microsoft, กลุ่มผู้ใช้ เช่น Share (กลุ่มผู้ใช้ IBM เมนเฟรม) และกลุ่มผู้พัฒนารวมอยู่ด้วย ทำให้เริ่มดูเหมือนว่า Unicode จะเป็นต่อในการแข่งขันนี้

ในปีนี้ กลุ่มผู้พัฒนารหัส Unicode ได้ร่วมกันจัดตั้งบริษัท Unicode ขึ้น ซึ่งนอกจากจะมีบริษัทดังกล่าวไว้ข้างต้นเป็นสมาชิกแล้ว ยังมีบริษัทอื่นเข้าร่วมด้วย ได้แก่บริษัท Aldus, บริษัท Go Corp., Lotus และบริษัท Novell และได้ออกแบบ รหัส Unicode 1.0 ออกมาเป็นผลสำเร็จ

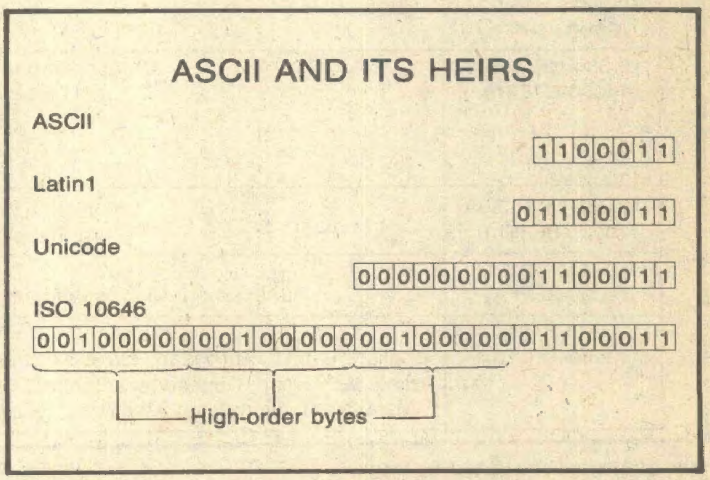
จากรูปที่ 1 แสดงถึงการเปรียบเทียบขนาดของรหัสมาตรฐานเดิมกับรหัสใหม่

เป้าหมายเริ่มแรกของรหัส Unicode

ภายหลังจากการเสนอแนวคิดของรหัส Unicode ออกมา Ken Whistler จากบริษัท Metaphor (และเลขาธิการบริษัท Unicode) ได้ร่างเป้าหมายในการออกแบบแสดงต่อกลุ่ม Share 76 ในซานฟรานซิสโก ในเป้าหมายดังกล่าวได้รวมเอาสิ่งต่อไปนี้ไว้ด้วยคือ

- ความสมบูรณ์

Unicode มีความตั้งใจที่จะครอบคลุมตัวอักษรทุกตัว แม้แต่ตัวอักษรในภาษาที่ตายแล้ว เช่น ภาษาสันสกฤต



รูปที่ 1 แสดงการเปรียบเทียบขนาดของรหัสมาตรฐานเดิมกับรหัสใหม่

และภาษาเก่าแก่อื่น ๆ ที่นับวันจะมีใช้น้อยลงทุกที

– ความง่ายและมีประสิทธิภาพ

ตัวอักษรทุกตัวในรหัส Unicode จะมีขนาด 16 บิตเท่ากันหมด จะไม่มีตัวควบคุมและลำดับหลีก (escape sequences) รวมอยู่ด้วย เพื่อไม่ให้เกิดความสับสนและความยากในการส่งข้อมูล

– ความไม่คลุมเครือ

รหัสแต่ละตัวใช้แทนตัวอักษรตัวเดียว จึงไม่ก่อให้เกิดปัญหาความผิดพลาดในการตีความหมายตัวอักษรที่ส่งตามมาภายหลัง

– ความถูกต้อง

รหัสแต่ละตัวได้รับการพิจารณาจากผู้เชี่ยวชาญด้านภาษาแล้วว่าเหมาะสม

– ความเที่ยงตรง

ในการแปลงผันข้อมูลที่เป็นตัวอักษรภาษากับรหัสมาตรฐานเดิม ข้อมูลจะไม่มีการสูญหายหรือผิดเพี้ยนไป

แม้เป้าหมายดังกล่าวจะไม่ได้เป็นไปตามที่วางไว้ทั้งหมด แต่ก็ทำให้เกิดเป็นรหัส Unicode ขนาด 16 บิต ซึ่งได้เตรียมที่ไว้สำหรับตัวควบคุมซึ่งเคยมีอยู่ในรหัสแอสกีด้วย เพื่อให้สามารถเข้ากันได้กับรหัสแอสกีเดิม แม้ว่าตัวควบคุมดังกล่าวจะไม่ใช้ในรหัส Unicode ก็ตาม และในรหัส Unicode นี้ ได้แยกความแตกต่างระหว่างรหัสและรูปแบบของตัวอักษรออกจากกัน โดยรูปแบบของตัวอักษรอาจต่างกันได้ แต่ต้องยังคงความหมายถึงตัวอักษรตัวเดียวกันดังรูปที่ 2

ในรูปที่ 2 แสดงรหัส Unicode ที่ได้รับการกำหนดไว้ โดยมี 8192 รหัสแรกใช้แทนตัวอักษรภาษาต่าง ๆ รวมทั้งที่ว่างสำหรับตัวอักษรของบางภาษา (ซึ่งมีดอกจันกำกับอยู่ด้วยในรูป) ที่จะได้รับการบรรจุลงไปภายหลัง เนื่องจากการออกแบบต้องการให้รวดเร็วทันการณ์ จึงจำเป็นต้องละการออกแบบตัวอักษรเหล่านี้เอาไว้ก่อน

สำหรับรหัส 4096 ตัวถัดมา จะใช้แทนเครื่องหมายวรรคตอน, ตัวกระทำทางคณิตศาสตร์, ตัวสัญลักษณ์ต่าง ๆ รวมทั้งตัวอักษรพิเศษอื่น ๆ ตามมาด้วยอีก 4096 ตัว สำหรับตัวอักษร และเครื่องหมายวรรคตอนในภาษาจีน, ญี่ปุ่น และเกาหลี ที่ไม่ใช่ตัวอักษรแบบคำ

ส่วนที่ใหญ่ที่สุดในรหัส Unicode คือ ส่วนถัดมาซึ่งมีขนาด 43,008 ตัว ซึ่งประกอบด้วยส่วนที่ใช้แทนตัวอักษรฮั่น 27,000 ตัว และที่ว่างสำหรับตัวอักษรชุดอื่น ๆ ที่อาจเพิ่มเติมเข้าไปภายหลัง นอกจากนี้ Unicode ยังได้เตรียมที่ 5632 ตัว เป็นส่วนสำหรับผู้ใช้ได้กำหนดรหัสขึ้นใช้เอง และส่วนของความเข้ากันได้ขนาด 495 ตัว สำหรับช่วยเหลือผู้พัฒนาในการแปลงรหัสของตนเป็นรหัส Unicode

กลายเป็น 2 มาตรฐาน

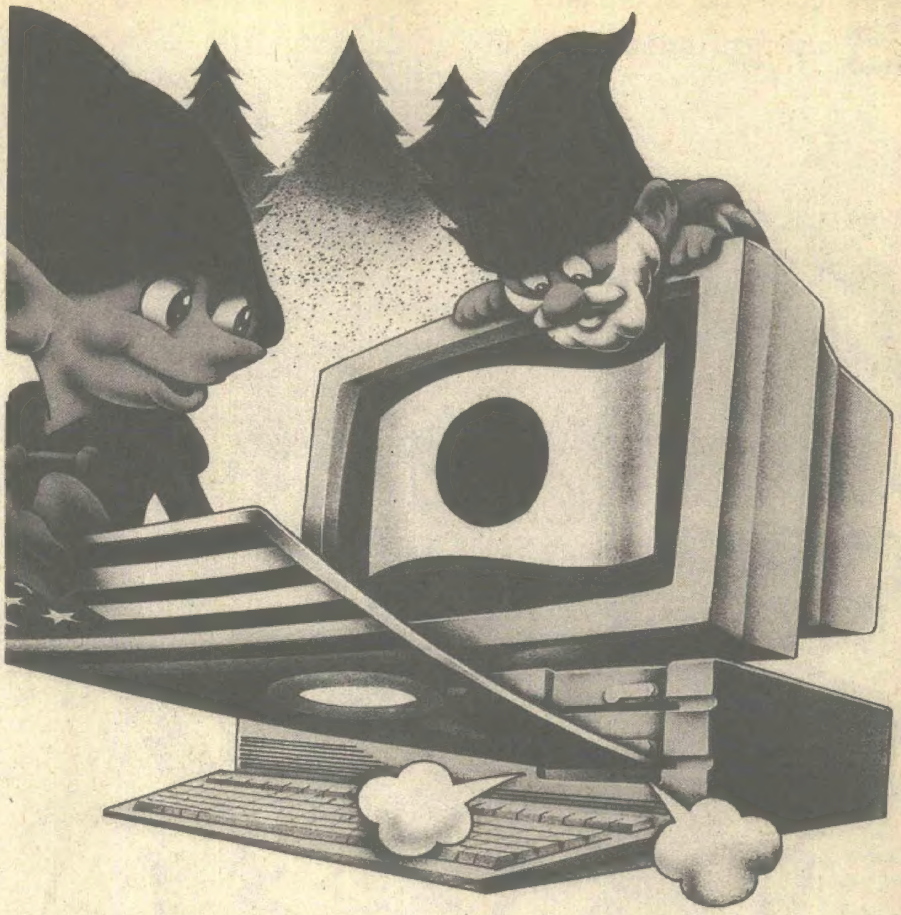
การเกิดขึ้นของรหัส Unicode ทำให้ในปัจจุบันมีรหัสมาตรฐานสำหรับหลายภาษาอยู่ 2 รหัส โดยมีข้อแตกต่างที่สำคัญบางประการดังนี้

– รหัส ISO 10646 ใช้ 4 ไบต์ เพื่อแทนตัวอักษร 1 ตัว (แต่มีวิธีการ Compaction ที่สามารถส่งข้อมูลเพียง

THE MANY CHARACTERS OF UNICODE

Codes	Characters	Examples
65,025 to 65,536 (FE00-FFFFh)	Compatibility area	
59,392 to 65,024 (E800-FDFFh)	User area	
16,384 to 59,391 (4000-E7FFh)	Chinese/Japanese/Korean ideographs Unified Han characters Future use	丰 毛
12,288 to 16,383 (3000-3FFFh)	Chinese/Japanese/Korean auxiliary Alphabets and punctuation	は の ㄴ
8192 to 12,287 (2000-2FFFh)	Symbols Punctuation, Mathematical operators, technical symbols, dingbats	; π ©
0 to 8191 (0000-1FFFh)	Alphabets English, Latin1, European, Extended Latin, Phonetic, Greek, Cyrillic, Armenian, Hebrew, Arabic, Ethiopian, Devanagari, Bengali, Gurmukhi, Gujarati, Oriya, Tamil, Telugu, Kannada, Malayalam, Sinhalese, Thai, Lao, Burmese*, Khmer*, Tibetan, Mongolian*, Georgian	a ë ¥

รูปที่ 2 ลักษณะของโค้ดหลาย ๆ แบบ



1, 2 หรือ 3 ไบต์ได้) ส่วนรหัส Unicode จะกำจัดตัวอักษรอันที่ซ้ำกันออกไป

– รหัส ISO 10646 เตรียมที่ 28,672 ตัวไว้สำหรับตัวควบคุมที่เคยมีในรหัสมาตรฐานเก่าทั้งหมด แต่รหัส Unicode เตรียมที่ไว้เพียง 65 ตัว สำหรับตัวควบคุมในรหัสแอสกีเดิมเท่านั้น

– รหัส ISO 10646 ได้เตรียมที่ไว้สำหรับตัวเน้นเสียง (accented letters) ซึ่งในรหัส Unicode ก็มีการกำหนดเอาไว้เช่นกัน แต่ยังได้เตรียมที่สำหรับตัวอักษรที่ประกอบตัวเน้นเสียงแล้วไว้ด้วย

นอกจากนี้ ยังมีข้อแตกต่างในการจัดการกับภาษา ซึ่งมี 2 ทิศทาง (ซ้ายไปขวาและขวามาซ้าย) ต่างกันด้วย เช่น ในภาษาฮีบรู และภาษาอารบิคที่เขียนจากขวามาซ้าย

ความแตกต่างของ 2 มาตรฐานนี้ทำให้เกิดปัญหาขึ้นบ้าง เนื่องจากความเห็นที่ไม่ตรงกันในการเข้ารหัสตัวอักษร ซึ่ง Ken Whistler ได้กล่าวได้ว่า "มันไม่เพียงพอที่จะกำหนดตัวเลขให้กับตัวอักษรแต่ละตัวเท่านั้น ถ้าเป็นเมื่อก่อนก็อาจจะกระทำได้กับตัวอักษรจำนวนน้อย ๆ แต่เป็นไปไม่ได้เมื่อคุณกำลังพยายามที่จะพัฒนารหัสมาตรฐานสากลเพื่อที่จะรองรับตัวอักษรทุกตัว" Whistler ยังเสริมไว้ด้วยว่า แม้แต่รหัสแอสกีที่ตายตัวและทุกคนเข้าใจตรงกัน ยังไม่มีมาตรฐานในการกำหนดการจบบรรทัด ซึ่งเป็นผลให้การส่งตัวปัดแคร่และตัวป้อนบรรทัดในระบบจดหมายอิเล็กทรอนิกส์ (E-mail) ที่มีการกำหนดลำดับของรหัสตัวควบคุมทั้งสองไว้ต่างกันอย่างตายตัว จะทำให้ข้อความที่ได้รับผิดไปจากเดิมที่ส่งมา ปัญหาที่สำคัญอีกอันหนึ่งก็คือ สถาปัตยกรรมคอมพิวเตอร์ที่จะมารองรับแต่ก็ยังเล็กน้อยเมื่อเทียบกับระบบการเข้ารหัสที่เปลี่ยนไปโดยสิ้นเชิง

Jerry Andersen ผู้ซึ่งเป็นทั้งโปรแกรมเมอร์อาวุโสของบริษัท ไอบีเอ็ม และประธานคณะกรรมการของสหรัฐอเมริกาที่เข้าร่วมพัฒนารหัส ISO 10646 ได้กล่าวไว้ว่า "การพัฒนา Unicode ได้เน้นให้เป็นรหัสสำหรับอนาคตโดยสิ้นเชิง แต่สำหรับ ISO 10646 ยังตระหนักถึงการเปลี่ยนแปลงระบบเดิมมาเป็นระบบที่รองรับรหัสใหม่ และความเข้ากันได้กับรหัสมาตรฐานเดิมด้วย"

ผลกระทบจากรหัสมาตรฐานใหม่

รหัสมาตรฐานใหม่ที่ใช้เนื้อที่หลายไบต์นี้จะมีผลต่ออุตสาหกรรมคอมพิวเตอร์อย่างไร ? กลุ่มผู้สนับสนุนทั้งรหัส Unicode และ ISO 10646 ได้คาดหวังเอาไว้ว่า รหัสมาตรฐานใหม่จะทำให้การขายระบบต่าง ๆ ให้กับนานาชาติทำได้ง่ายขึ้น การพัฒนาระบบให้สามารถใช้ได้กับหลายประเทศทำได้โดยเพียงแต่เปลี่ยนข้อความเป็นภาษาแต่ละภาษาเท่านั้น

ในส่วนของผู้พัฒนาส่วนชุดคำสั่งระบบ การรับรหัสมาตรฐานใหม่เข้ามาใช้หมายถึง การที่จำเป็นจะต้องปรับปรุงชุดคำสั่งระบบในบางส่วน แม้ว่าการเปลี่ยนซอฟต์แวร์ต่าง ๆ ให้ใช้กับรหัสใหม่ได้อาจจะเสียค่าใช้จ่ายสูง แต่การกระทำดังกล่าวก็มีความจำเป็นเฉพาะในองค์กรที่มีความต้องการชุดตัวอักษรหลาย ๆ ภาษาเท่านั้น ในเรื่องนี้ Whistler ได้กล่าวเสริมไว้ว่า "Unicode เป็นระบบการเข้ารหัสที่ได้รับการออกแบบมาอย่างดี ซึ่งจะทำให้การพัฒนาซอฟต์แวร์บนรหัสนี้มีต้นทุนต่ำกว่าการเพิ่มชุดตัวอักษรที่ออกแบบมาไม่ดีเข้าไป" บริษัทหลายบริษัทที่เป็นสมาชิกของ Unicode ได้ประกาศออกมาแล้วว่า ในเวอร์ชันใหม่ของซอฟต์แวร์ของตน (ในจำนวนนี้มีส่วนชุดคำสั่งระบบของบริษัท Microsoft และ Apple รวมอยู่ด้วย) จะมีการเพิ่ม

COMPUTER STANDARDS

มาตรฐานใหม่ของรหัสแอสกี

การใช้รหัส Unicode เข้าไปด้วย ซึ่งสำหรับผู้ใช้ ถ้าไม่มีความต้องการที่จะใช้ตัวอักษรภาษาอื่น ๆ นอกจากตัวอักษรภาษาอังกฤษที่มีอยู่เดิม ก็ไม่มีความจำเป็นที่จะต้องเปลี่ยนไปใช้เวอร์ชันใหม่ของซอฟต์แวร์เหล่านี้

ในส่วนของฮาร์ดแวร์ ขนาดของรหัสที่ใหญ่ขึ้นจำเป็นต้องใช้เนื้อที่แรมและฮาร์ดดิสก์เพิ่มขึ้นมิใช่หรือ ? คำตอบคือ ใช่ แต่ก็ไม่ได้มากมายอย่างที่เราคิดกันเสียทีเดียว คือจะมีการใช้เนื้อที่แรมเพิ่มขึ้นจากเดิมเพียงประมาณ 10 ถึง 20 เปอร์เซ็นต์ ในเรื่องนี้ Bud Tribble กล่าวว่า "ถ้าคุณใช้แรมขนาด 8 เมกะไบต์ และเดิมใช้เนื้อที่ในแรม 7 เมกะไบต์ การเปลี่ยนไปใช้รหัส Unicode อาจเพิ่มการใช้เนื้อที่ในแรมขึ้นเป็น 7.7 เมกะไบต์ ซึ่งไม่ทำให้เกิดความจำเป็นที่จะต้องเพิ่มขนาดของแรม" แน่นอน ถ้าเป็นแฟ้มข้อความย่อมจะมีขนาดเพิ่มขึ้นเท่าตัว แต่ในปัจจุบัน แฟ้มที่เป็นข้อความล้วน ๆ มีจำนวนน้อยลง และแฟ้มข้อความที่เก็บรวมกับรูปภาพแบบกราฟิกมีจำนวนเพิ่มขึ้น ซึ่งในกรณีหลังขนาดของแฟ้มจะเพิ่มขึ้นเพียงประมาณ 20 ถึง 25 เปอร์เซ็นต์ นอกจากนี้ ราคาของหน่วยเก็บยังถูกลงเรื่อย ๆ ที่สำคัญที่สุดคือ เมื่อเทียบกับผลกำไรที่จะได้รับเพิ่มขึ้นจากการใช้รหัสมาตรฐานหลายภาษานี้แล้ว จะทำให้ค่าใช้จ่ายในการปรับปรุงระบบไร้น้ำหนักไปโดยสิ้นเชิง

ยกตัวอย่างข้อได้เปรียบของรหัสใหม่เช่น ด้วยมาตรฐานที่ตรงกัน เราสามารถจะส่งจดหมายอิเล็กทรอนิกส์ไปยังประเทศญี่ปุ่นได้อย่างมั่นใจว่า จะไม่เกิดการผิดเพี้ยนของข้อความและเรายังสามารถที่จะโอนย้ายข้อมูลระหว่างเครื่องแมคอินทอชกับเครื่องพีซีได้ โดยไม่ต้องทำการเปลี่ยนแปลงตัวอักษรเลย

มี 2 มาตรฐานไม่ดีแน่ จะแก้ไขอย่างไร

ได้มีการสำรวจความเห็นที่มีต่อรหัส ISO 10646 ในสหรัฐอเมริกา และมีรายงานที่ผ่านมาว่า เสียงส่วนใหญ่ไม่ให้การสนับสนุนรหัส 10646 คณะกรรมการพัฒนารหัส 10646 จึงได้เสนอแนวทางในการแก้ปัญหาไว้ 2 ประการ ประการแรกคือ การรับรหัส Unicode เข้าไว้เป็นส่วนหนึ่งของ 10646 และประการที่สองคือ การปรับปรุงรหัส 10646 ในส่วนที่มีข้อโต้แย้งที่สำคัญ

บทสรุป

การมี 2 มาตรฐานในการเข้ารหัสตัวอักษรทุก ๆ ภาษาย่อมไม่ดีแน่ คงไม่มีผู้ใดต้องการให้เป็นเช่นนั้น แต่สภาพการณ์ในปัจจุบันดูเหมือนว่า การมี 2 รหัสมาตรฐานเป็นสิ่งที่หลีกเลี่ยงไม่ได้เสียแล้ว สิ่งที่ทำได้ในตอนนี้ คือความร่วมมือกันของผู้พัฒนาในอันที่จะทำให้รหัสทั้งสองมีความเหมือนกันมากที่สุด ซึ่งพอจะมีแนวทางแล้วว่าอาจจะมีการรับเอารหัส Unicode เข้าไปเป็นส่วนหนึ่งของรหัส ISO 10646 ซึ่งจะทำให้เสมือนกับการมีมาตรฐานเดียว ภายในไม่เกินกึ่งทศวรรษนี้ เราคงจะมีผลิตภัณฑ์ที่เข้ามารองรับรหัสมาตรฐานใหม่นี้ได้ใช้กันเป็นจำนวนมากทีเดียว